READ PORT/PIPE

WRITE PORT/PIPE

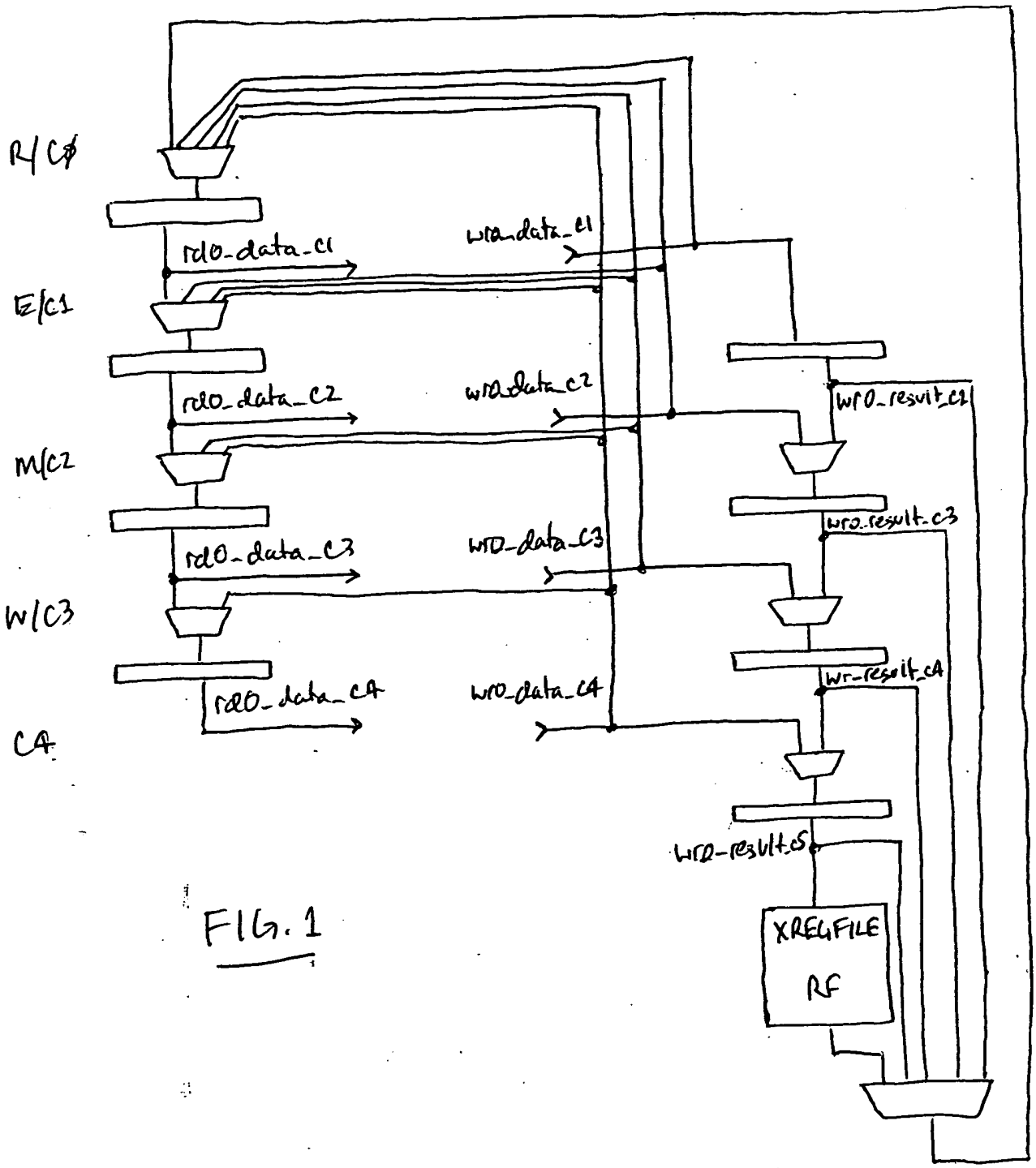


FIG. 1

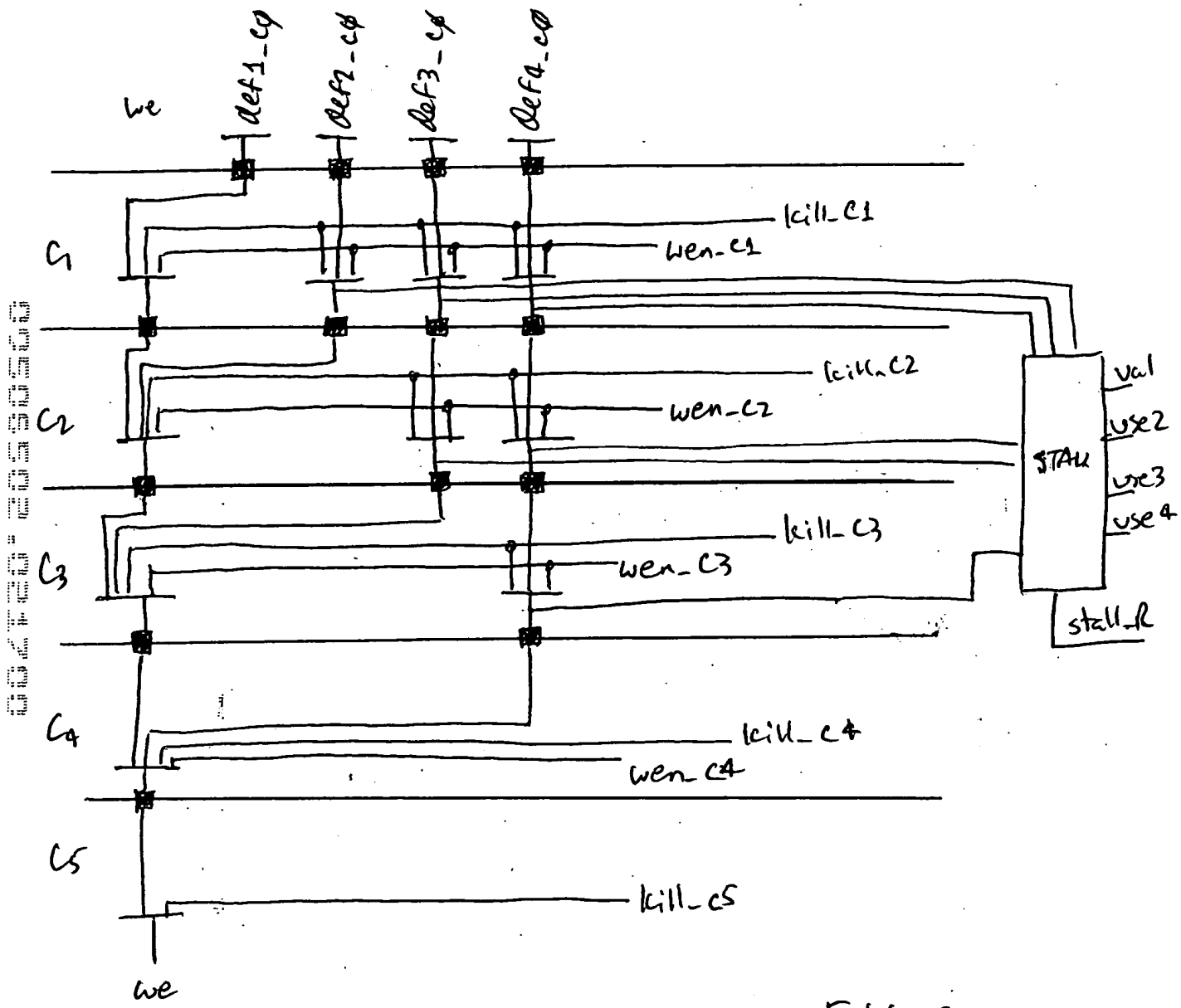

we
def1_c0
def2_c0
def3_c0
def4_c0
kill_c1
wen_c1
c1
kill_c2
wen_c2
c2
kill_c3
wen_c3
c3
kill_c4
wen_c4
c4
kill_c5
c5
we
STALL
val
use2
use3
use4
stall_R

FIG. 2

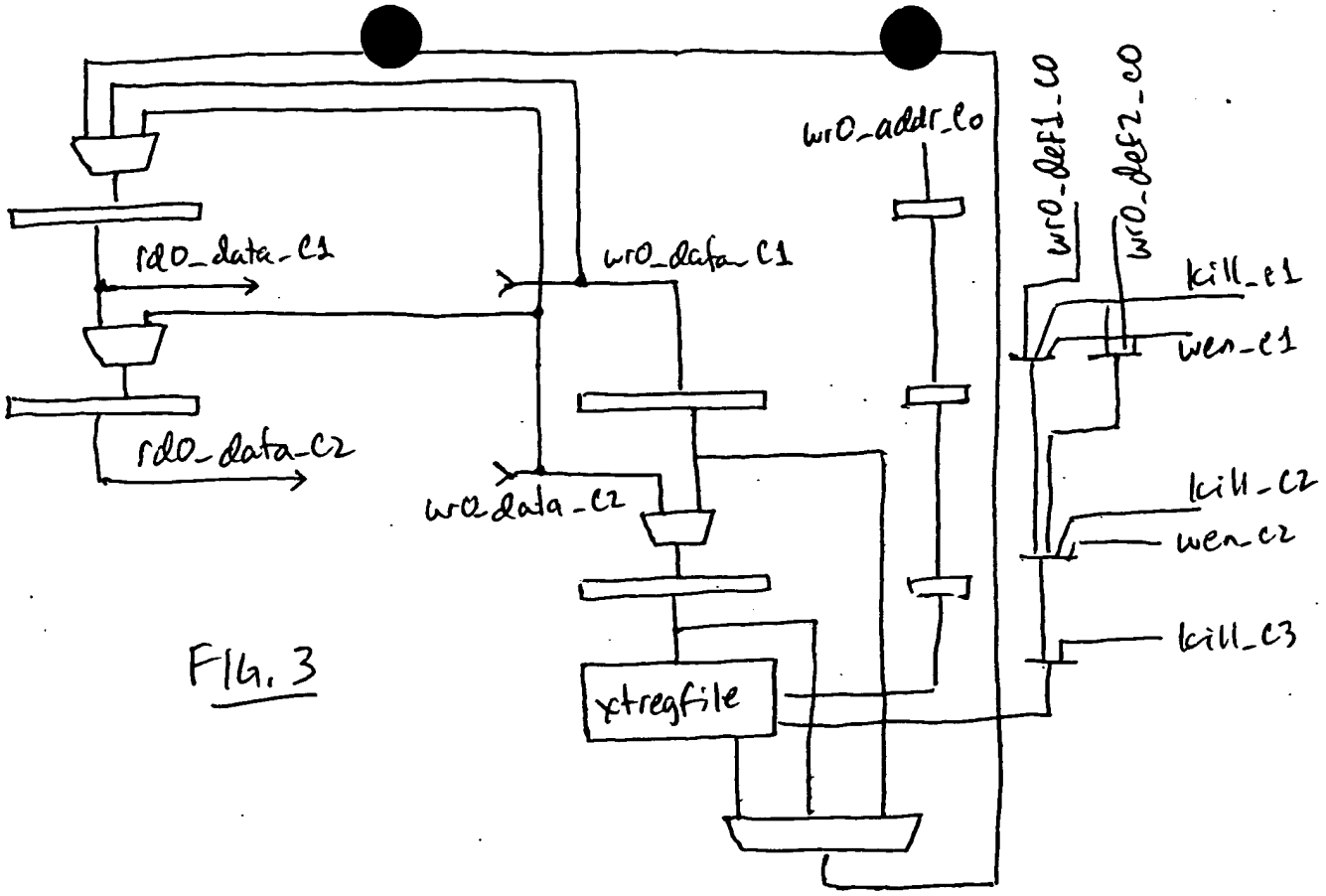

wr0_addr_c0
wr0_def1_c0
wr0_def2_c0
rd0_data_c1
wr0_data_c1
kill_c1
wen_c1
rd0_data_c2
wr0_data_c2
kill_c2
wen_c2
kill_c3
xtregfile

FIG. 3

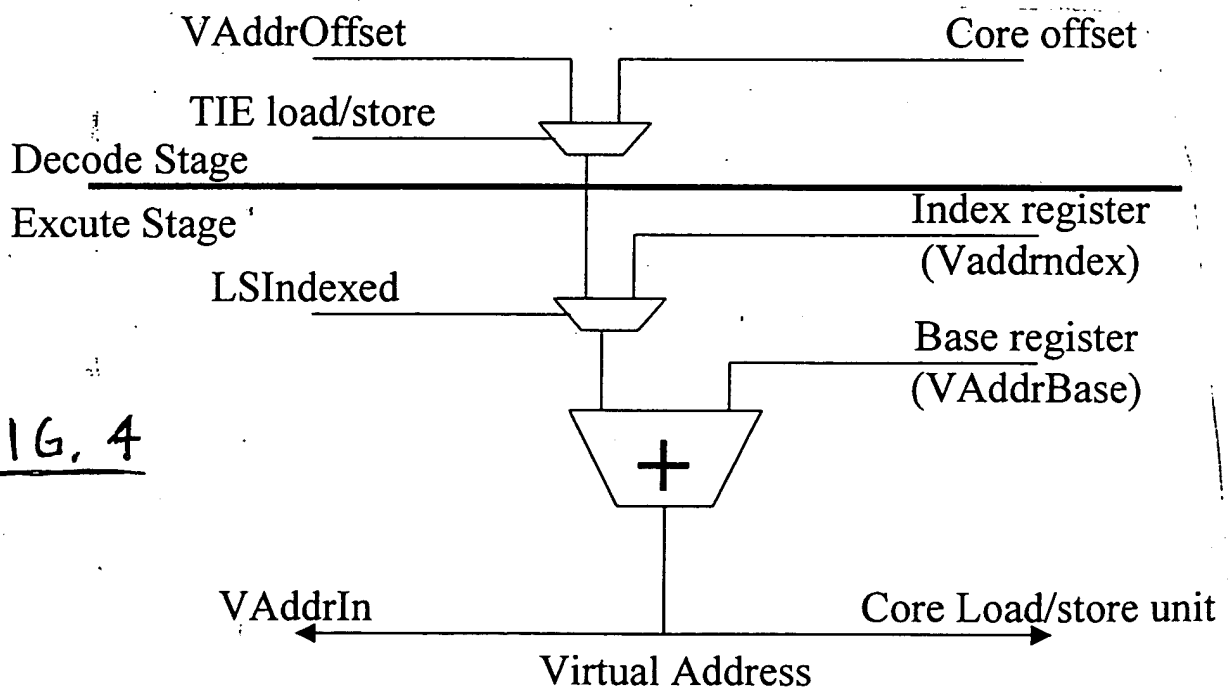

VAddrOffset
Core offset
TIE load/store
Decode Stage
Excute Stage
Index register
(Vaddrndex)
LSIndexed
Base register
(VAddrBase)
+
VAddrIn
Core Load/store unit
Virtual Address

FIG. 4

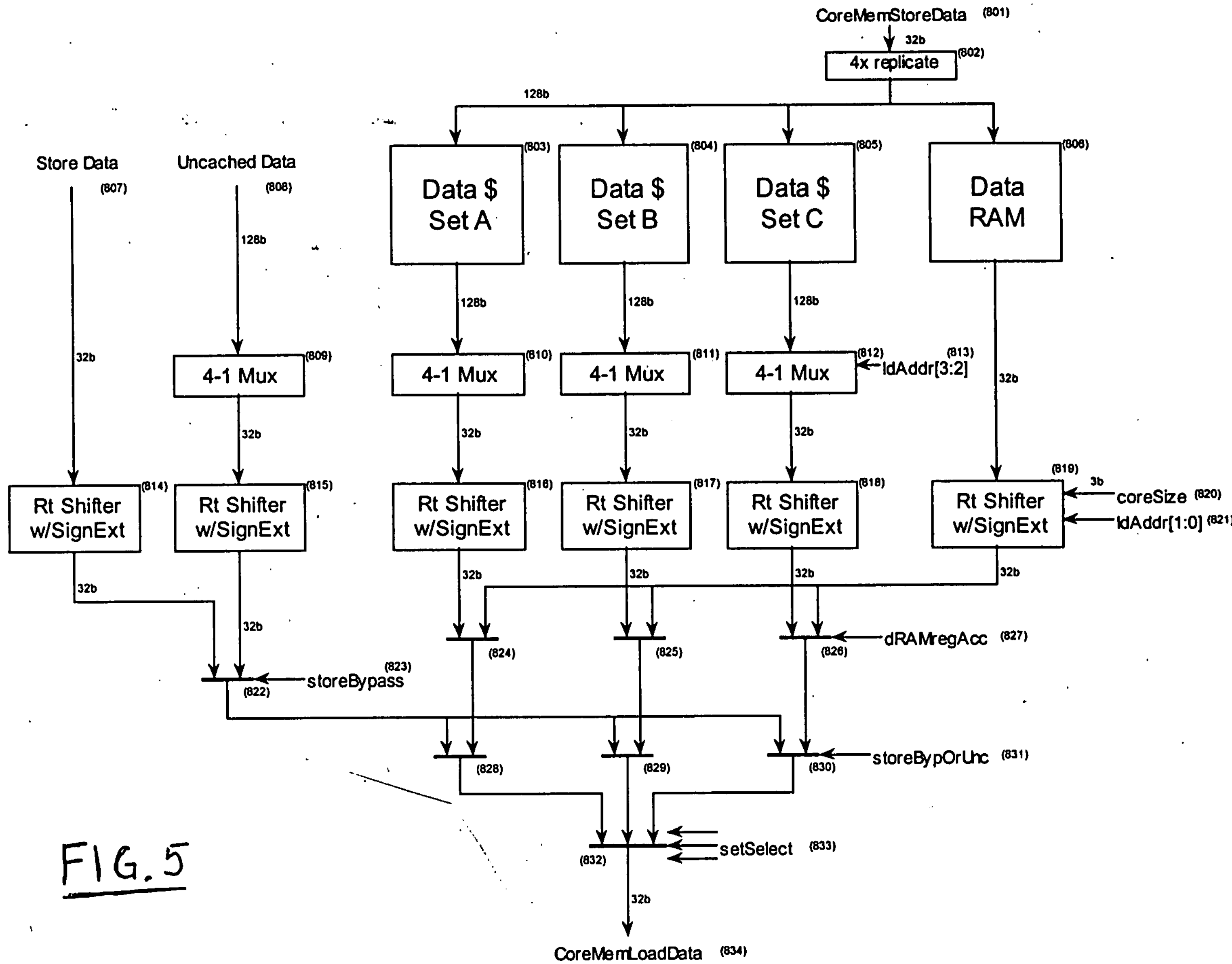

CoreMemStoreData (801)
32b
4x replicate (802)
128b
Data $ Set A (803)
Data $ Set B (804)
Data $ Set C (805)
Data RAM (806)
Store Data (807)
Uncached Data (808)
128b
4-1 Mux (809)
4-1 Mux (810)
4-1 Mux (811)
4-1 Mux (812)
ldAddr[3:2] (813)
32b
Rt Shifter w/SignExt (814)
Rt Shifter w/SignExt (815)
Rt Shifter w/SignExt (816)
Rt Shifter w/SignExt (817)
Rt Shifter w/SignExt (818)
Rt Shifter w/SignExt (819)
3b
coreSize (820)
ldAddr[1:0] (821)
(822)
storeBypass (823)
(824)
(825)
(826)
dRAMregAcc (827)
(828)
(829)
(830)
storeBypOrUnc (831)
(832)
setSelect (833)
CoreMemLoadData (834)

FIG. 5

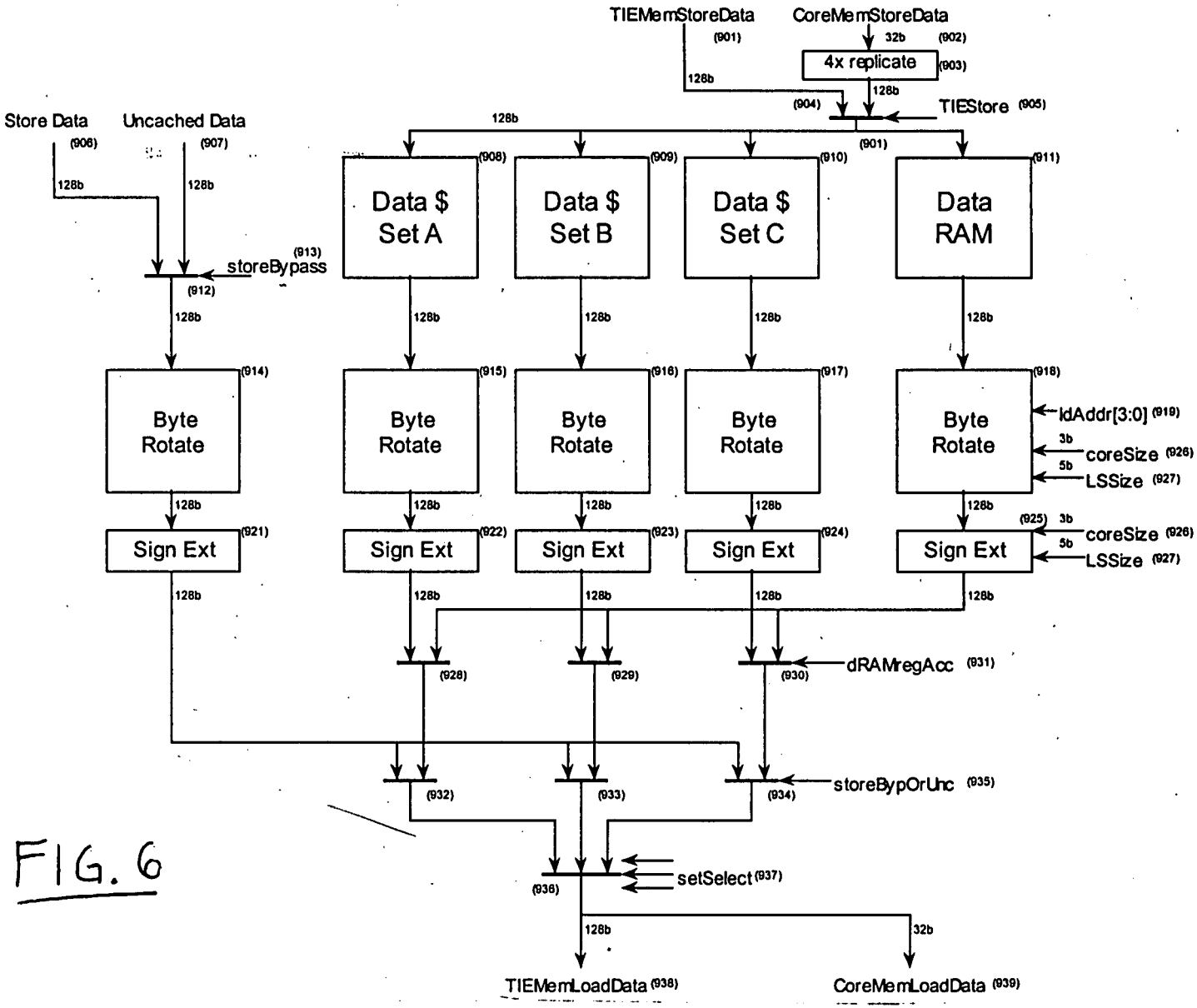


FIG. 6

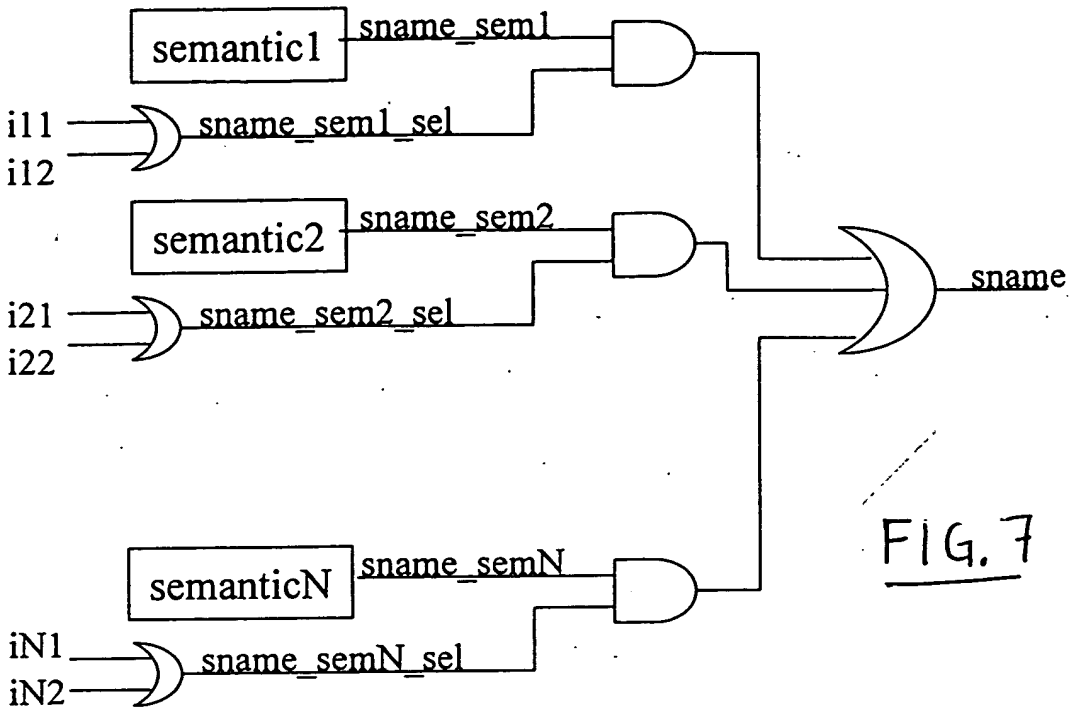

semantic1
sname_sem1
i11
i12
sname_sem1_sel
semantic2
sname_sem2
i21
i22
sname_sem2_sel
sname
semanticN
sname_semN
iN1
iN2
sname_semN_sel

FIG. 7

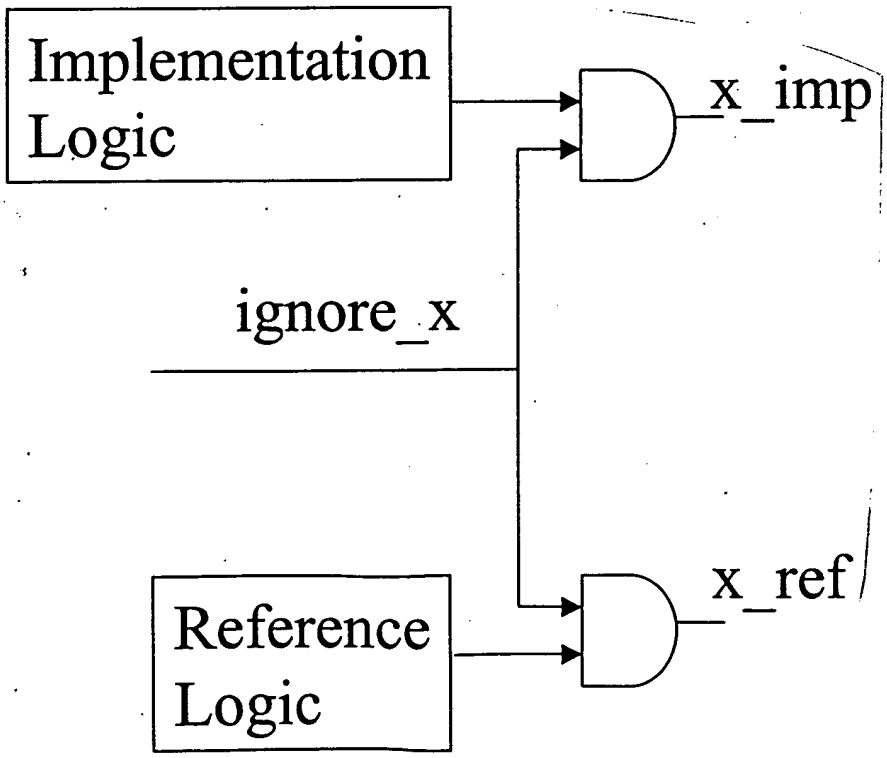

Implementation Logic
x_imp
ignore_x
Reference Logic
x_ref

FIG. 11

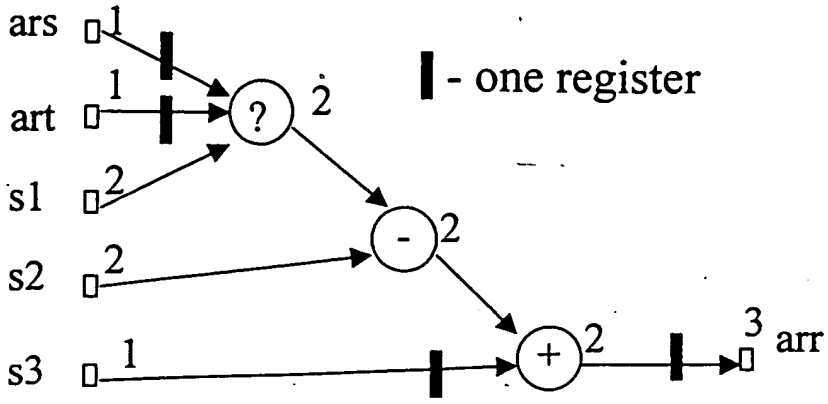


FIG. 8(a)

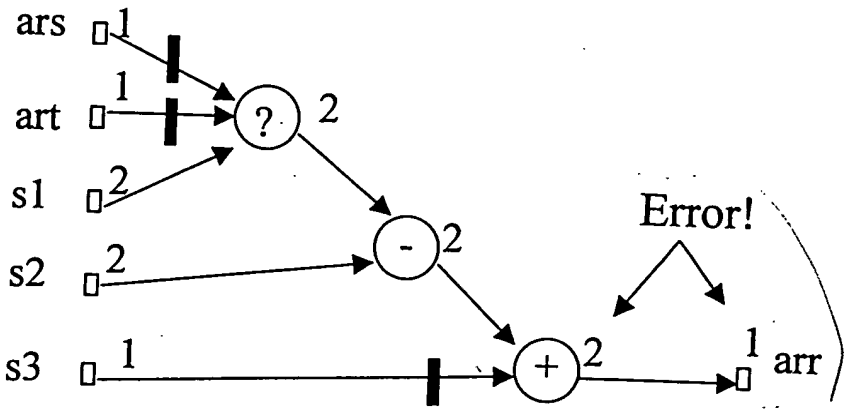


FIG. 8(b)

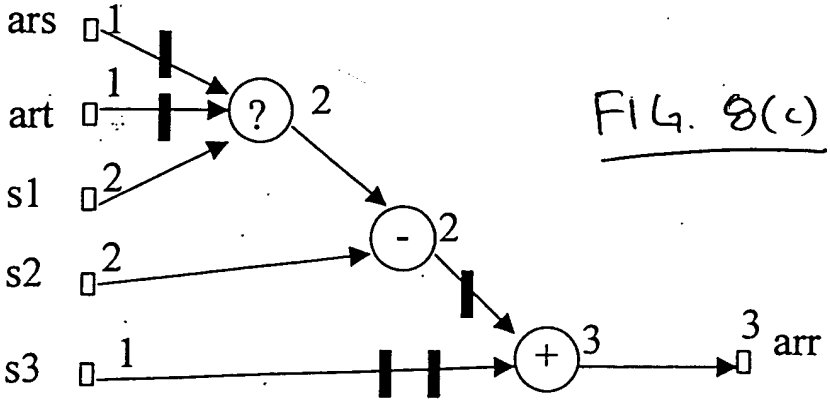


FIG. 8(c)

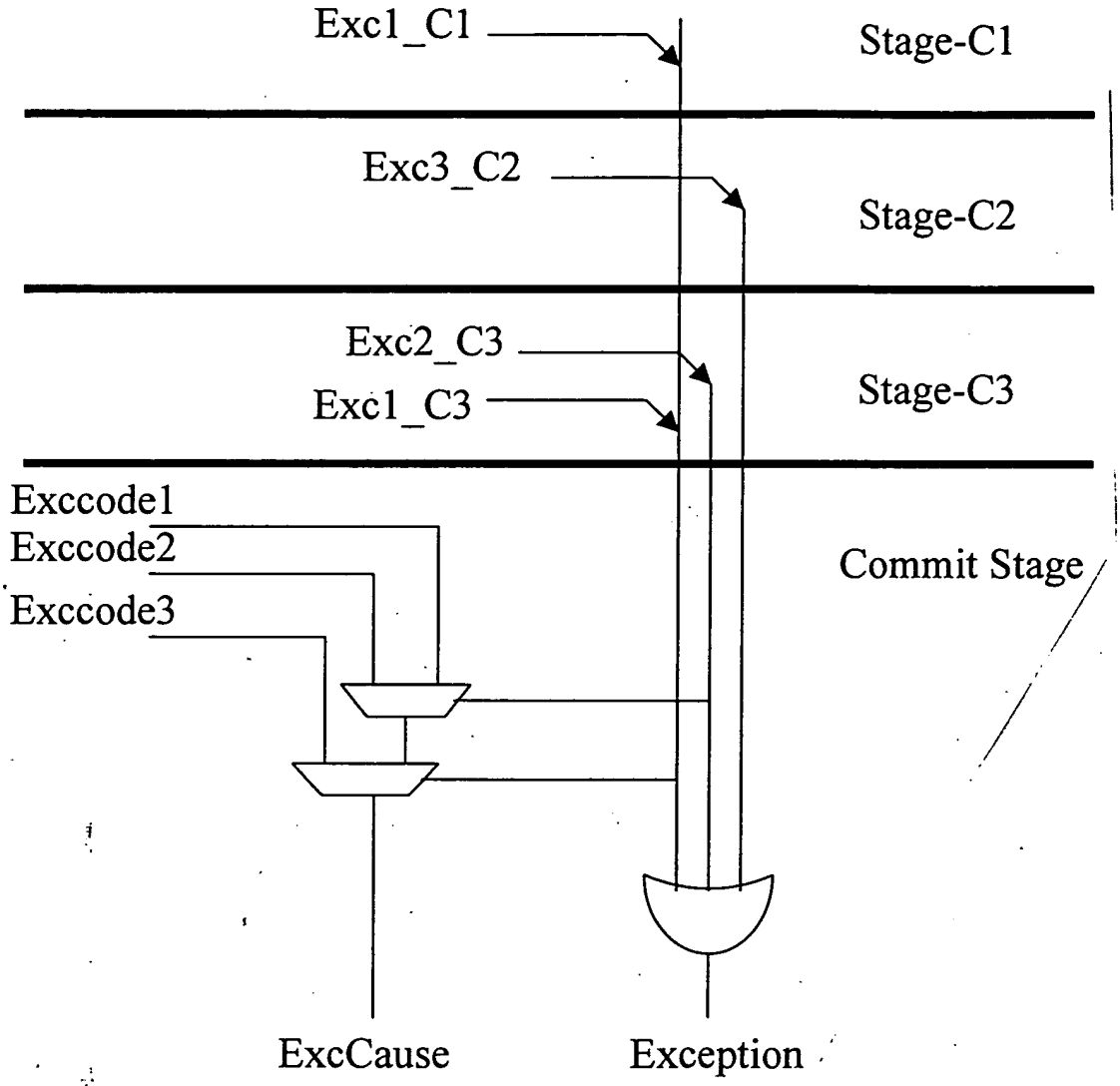

Exc1_C1
Stage-C1
Exc3_C2
Stage-C2
Exc2_C3
Exc1_C3
Stage-C3
Exccode1
Exccode2
Exccode3
Commit Stage
ExcCause
Exception

FIG. 9

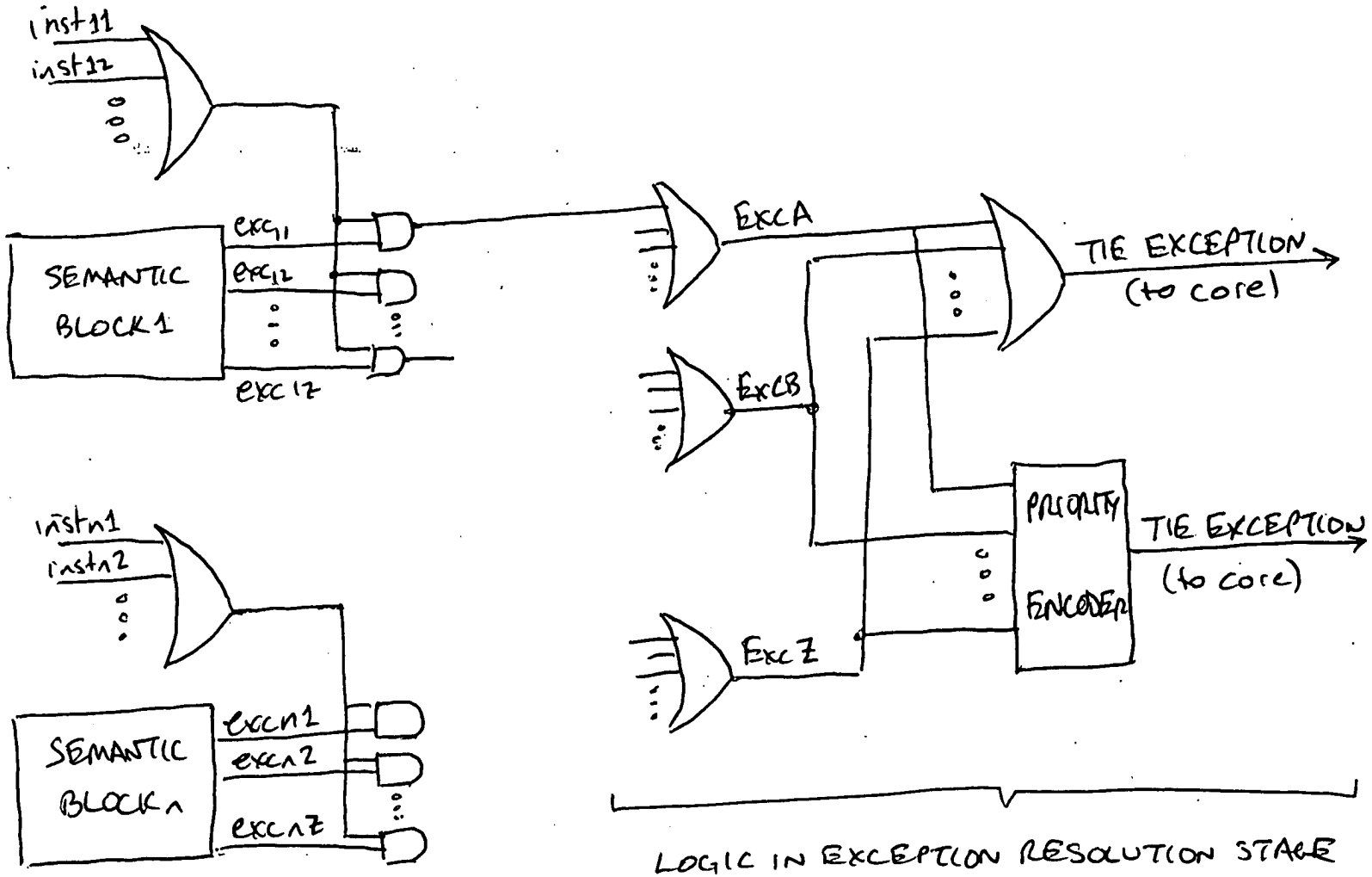
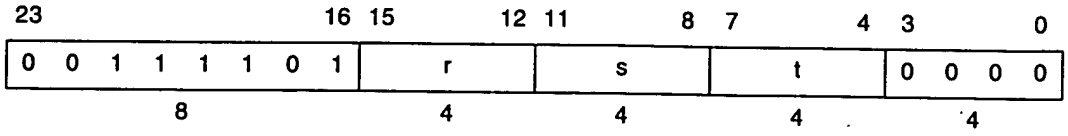

inst11
inst12
SEMANTIC BLOCK 1
exc11
exc12
exc1z
instn1
instn2
SEMANTIC BLOCKn
excn1
excn2
excnz
EXCA
EXCB
EXCZ
PRIORITY ENCODER
TIE EXCEPTION (to core)
TIE EXCEPTION (to core)
LOGIC IN EXCEPTION RESOLUTION STAGE

FIG. 10

MAX40

4 Parallel 40-bit Maximum

Instruction Word



Package

Vector Integer Coprocessor

Assembler Syntax

```
MAX40 vr, vs, vt
```

Description

MAX40 calculates the 40-bit two's complement maximum value for each of the 4 elements of vector registers vs and vt. The result elements are written to vector register vr.

Operation

```
    vr = ((({~vs[159], vs[158:120]}) < ({~vt[159], vt[158:120]})) ?
vt[159:120] : vs[159:120], (({~vs[119], vs[118:80]}) < ({~vt[119],
vt[118:80]})) ? vt[119:80] : vs[119:80], (({~vs[79], vs[78:40]}) <
({~vt[79], vt[78:40]})) ? vt[79:40] : vs[79:40], (({~vs[39],
vs[38:0]}) < ({~vt[39], vt[38:0]})) ? vt[39:0] : vs[39:0]};
```

Exceptions

None

FIG. 12

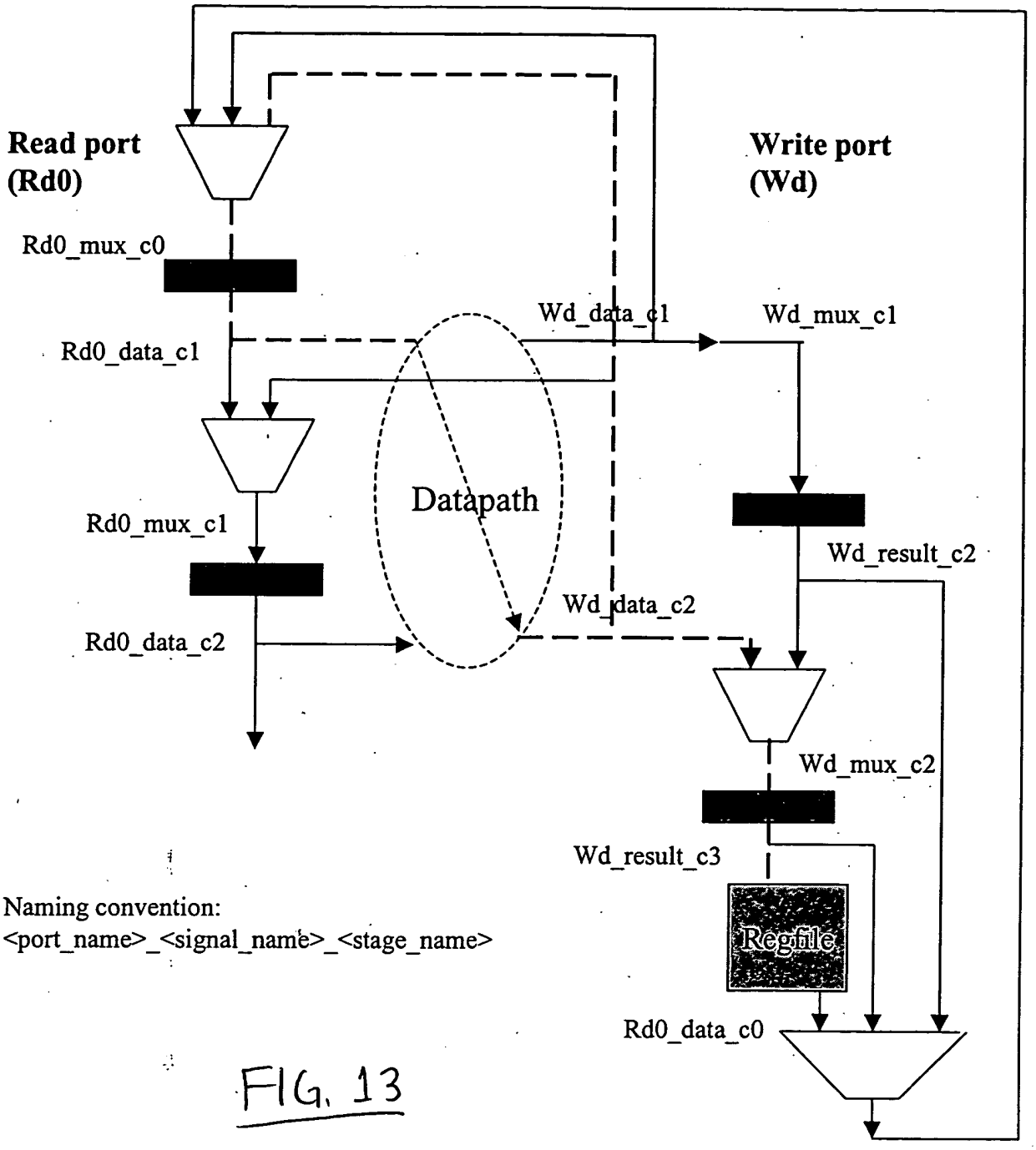

Read port
(Rd0)
Write port
(Wd)
Rd0_mux_c0
Rd0_data_c1
Wd_data_c1
Wd_mux_c1
Datapath
Rd0_mux_c1
Wd_result_c2
Wd_data_c2
Rd0_data_c2
Wd_mux_c2
Wd_result_c3
Regfile
Rd0_data_c0
Naming convention:
<port_name>_<signal_name>_<stage_name>

FIG. 13

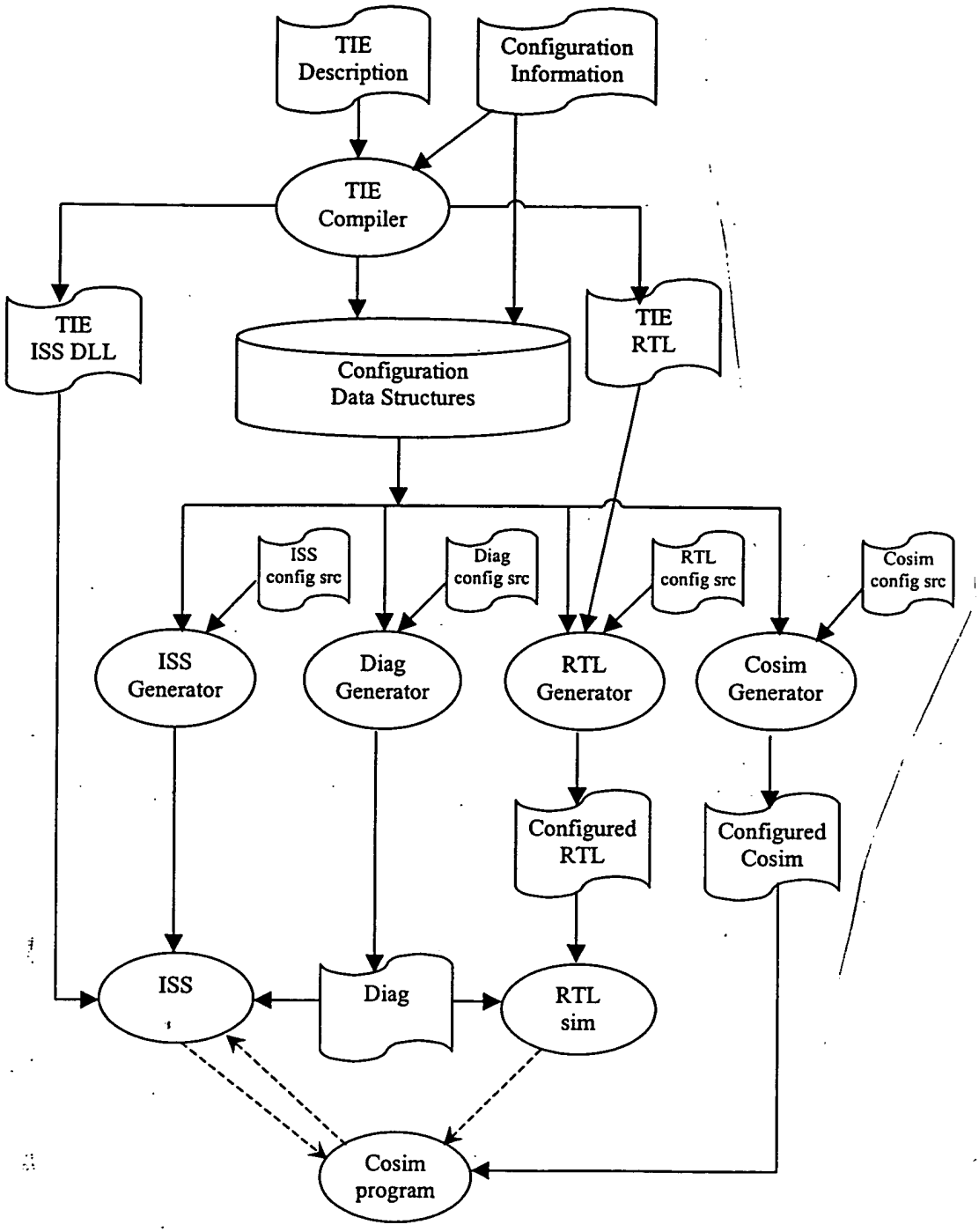

TIE Description
Configuration Information
TIE Compiler
TIE ISS DLL
Configuration Data Structures
TIE RTL
ISS config src
Diag config src
RTL config src
Cosim config src
ISS Generator
Diag Generator
RTL Generator
Cosim Generator
Configured RTL
Configured Cosim
ISS
Diag
RTL sim
Cosim program

FIG. 14